

Robust PCA and subspace tracking from incomplete observations using ℓ_0 -surrogates*

Clemens Hage and Martin Kleinsteuber

Department of Electrical Engineering and Information Technology
Technische Universität München
Arcisstr. 21, 80333 Munich, Germany
{hage,kleinsteuber}@tum.de
<http://www.gol.ei.tum.de>

October 3, 2012

Abstract

Many applications in data analysis rely on the decomposition of a data matrix into a low-rank and a sparse component. Existing methods that tackle this task use the nuclear norm and ℓ_1 -cost functions as convex relaxations of the rank constraint and the sparsity measure, respectively, or employ thresholding techniques. We propose a method that allows for reconstructing and tracking a subspace of upper-bounded dimension from incomplete and corrupted observations. It does not require any a priori information about the number of outliers. The core of our algorithm is an intrinsic Conjugate Gradient method on the set of orthogonal projection matrices, the so-called Grassmannian. Non-convex sparsity measures are used for outlier detection, which leads to improved performance in terms of robustly recovering and tracking the low-rank matrix. In particular, our approach can cope with more outliers and with an underlying matrix of higher rank than other state-of-the-art methods.

1 Introduction

The detection of subspaces that best fit high-dimensional data is a challenging and important task in data analysis with countless applications. The classic Principal Component Analysis (PCA) is still the standard tool for searching the best rank- k approximation of a high-dimensional data set $X \in \mathbb{R}^{m \times n}$, where the approximation quality is measured in terms of the Frobenius norm. On the one hand, this minimization problem can be solved easily via a Singular Value Decomposition (SVD), while on the other hand the choice of the Frobenius norm makes this approach highly vulnerable to heavy outliers. In the past years a lot of effort has been spent on methods that allow to find the best fitting subspace despite the presence of heavy outliers in the measurements or missing data. Certain approaches in this area commonly known as *Robust PCA* stick closely to the classic PCA concept and robustify it by measuring the distance from the

*This work has been supported by the DFG excellence initiative research cluster *CoTeSys*.

data points to the subspace with ℓ_1 -cost functions, cf. Ding et al (2006), Kwak (2008). Others, including the approach presented here, relate Robust PCA to the field of robust matrix completion. That is, given some incomplete observations \hat{X} of a corrupted and possibly noisy data matrix X , which is the superposition of a low-rank matrix L and a sparse matrix S , the task is to recover L and S .

1.1 Related Work

One of the most prominent approaches to Robust PCA in the past years is proposed by Candes et al (2009) and Wright et al (2009). The authors formulate a convex optimization problem by relaxing the hard rank constraint to a nuclear norm minimization and analyze how well the ℓ_1 -relaxation approximates the ℓ_0 -norm in the low-rank and sparse decomposition task. Methods to solve the problem include Singular Value Thresholding (SVT, Cai et al (2010)) and the exact (EALM) or inexact (IALM) augmented Lagrangian multiplier method (Lin et al, 2010), which are fast and comparably reliable if the underlying assumptions, i.e. low-rank-property and sparsity of outliers, are valid. The problem of reconstructing the low-rank matrix L in the case where entire columns in the measurements are corrupted is considered by Chen et al (2011). A method called *SpaRCS* is proposed by Waters et al (2011), which recovers the low-rank and the sparse part of a matrix from compressive measurements.

In many applications it is reasonable to assume that no additive Gaussian noise is present in the model, as noise is negligible compared to the signal power, e.g. when dealing with high-quality sensors in the field of Computer Vision. However, there also exist approaches such as *GoDec* (Zhou and Tao, 2011) that explicitly model additional Gaussian noise. The method uses thresholding techniques and extends the low-rank and sparse decomposition to the noisy case. *SpaRCS* and *GoDec* aim at recovering both the low-rank matrix L and the outliers S from noisy measurements and therefore require an upper bound for the cardinality of the sparse component.

In contrast to the often-performed rank-relaxation, there also exist methods which fix the dimension of the subspace approximation, such as the greedy algorithm *GECO* (Shalev-Shwartz et al, 2011). This method reconstructs a dataset iteratively based on SVD while increasing the rank of the approximation in each step. A general framework for optimizing over matrices of a fixed rank has also been proposed by Shalit et al (2010). One drawback here is that the manifold requires a fixed rank k and optimal points that may have a rank strictly lower than k are not in the feasible set.

An alternative and elegant way to control the rank in terms of an upper bound is optimization on the set of fixed-dimensional subspaces, the so-called Grassmannian. As pointed out by Meyer et al (2011), optimizing on the Grassmannian offers many advantages, such as limited memory usage and a lower number of parameters to optimize. Although the optimization problem becomes non-convex, reliable performance can be achieved in practice. In the work of Keshavan and Montanari (2010), spectral techniques are combined with a learning algorithm on the Grassmannian. Boumal and Absil (2011) propose a Riemannian trust-region method on the Grassmannian for low-rank matrix completion, which, however, does not consider heavy outliers. The *GROUSE* algorithm (Balzano et al, 2010) furthermore demonstrates that optimization on the Grassmannian allows to estimate the underlying subspace incrementally. Instead of batch-processing a data set, the samples can be processed one at a time, which makes it possible to track a subspace that varies over time, even if it is incompletely observed. A small portion of the data is used to obtain an initial subspace estimate and with every new incoming data sample this estimate is modified to follow changes in the

dominant subspace over time. While the method of Balzano et al (2010) operates with ℓ_2 -cost functions, its recent adaptation *GRASTA* (He et al, 2012) performs a gradient descent on the Grassmannian and aims at optimizing an ℓ_1 -cost function to mitigate the effects of heavy outliers in the subspace tracking stage. The authors overcome the non-differentiability of the ℓ_1 -norm by formulating an augmented Lagrangian optimization problem at the cost of doubling the number of unknown parameters.

1.2 Our Contribution

Although the ℓ_1 -norm leads to favorably conditioned optimization problems it is well-known that penalizing with non-convex ℓ_0 -surrogates allows reconstruction even in the case when ℓ_1 -based methods fail, see e.g. Chartrand and Staneva (2008). Therefore, we propose a framework that combines the advantages of Grassmannian optimization with non-convex sparsity measures. Our approach focuses primarily on reconstructing and tracking the underlying subspace and can operate on both fully and incompletely observed data sets. The algorithm performs a low-rank and sparse decomposition. However, it does not require any information about the cardinality or the support set of the sparse outliers, thus being different from *SpaRCS* and *GoDec*.

In contrast to *GRASTA* (He et al, 2012), the method presented in this paper directly optimizes the cost function and thus operates with less than half the number of unknowns. Like all optimization methods on the Grassmannian our algorithm allows to upper-bound the dimension of the underlying subspace and easily extends to the problem of robustly tracking this subspace.

Experimental results confirm that the proposed method can cope with more outliers and with an underlying matrix of higher rank than other state-of-the-art methods, thus extending possible areas of application from the strict low-rank and sparse decomposition to the more general area of robust dimensionality reduction. In the following section we present an alternating minimization scheme and relate our approach to dimensionality reduction via PCA. Subsequently, we carefully derive and explain a Conjugate Gradient (CG) type algorithm on the Grassmannian for solving the individual minimization tasks. We then extend the static method by a dynamic subspace tracking algorithm and finally, we evaluate the performance of the proposed method with various ℓ_0 -surrogates and compare our approach to other state-of-the-art methods.

2 Problem statement

Let $X \in \mathbb{R}^{m \times n}$ be the data matrix of which we select the partial entries $\hat{X} = \mathcal{A}(X)$ using a linear operator. We consider the data model $X = L + S$ where L is of rank $\text{rk}(L) \leq k$ and S is sparse. Our aim is to recover L from the observations \hat{X} . A direct approach leads to the numerically infeasible optimization problem

$$\min_{\text{rk } L \leq k} \|\hat{X} - \mathcal{A}(L)\|_0. \quad (1)$$

Since $\text{rk}(L) \leq k$, we can write the matrix L as the product $L = UY$, where $Y \in \mathbb{R}^{k \times n}$ and U is an element of the so-called *Stiefel manifold* $\text{St}_{k,m} = \{U \in \mathbb{R}^{m \times k} | U^\top U = I_k\}$ with I_k denoting the $(k \times k)$ -identity matrix. This factorization of L has the following practical interpretation: U is an orthonormal basis of the robustly estimated subspace of the first k principal components, hereafter referred to as *(robust) dominant subspace*. Note, that neither U nor Y are uniquely determined in this factorization. Let $O(k) = \{\theta \in$

$\mathbb{R}^{k \times k} \mid \theta^\top \theta = I_k\}$ define the set of orthogonal matrices, then U can always be adjusted by an orthogonal matrix $\theta \in O(k)$, such that $UY = U\theta\theta^\top Y$ with $\theta^\top Y$ having uncorrelated rows, leading to the k (robustly estimated) principle component scores of X .

Our concession to the numerical infeasibility of (1) is to employ an alternative, possibly non-convex sparsity measure h . Some concrete examples can be found in Section 6.1. As a result, problem (1) relaxes to the minimization problem

$$\min_{U \in \text{St}_{k,m}, Y \in \mathbb{R}^{k \times n}} h(\hat{X} - \mathcal{A}(UY)). \quad (2)$$

Problem (2) can be addressed in two different ways. Either the full data set \hat{X} is processed at once, resulting in one estimate for U and Y , or the data vectors \hat{x} are processed one at a time. In the latter case, while for each new sample \hat{x} one obtains a corresponding optimal coordinate set y , the subspace estimate U should vary smoothly over time and fit both recent and current observations. We will refer to the former problem as *subspace reconstruction* or *Robust PCA* and to the latter as *subspace tracking*.

3 An alternating minimization framework using ℓ_0 surrogates

Directly tackling problem (2) has two severe drawbacks. The first is the above mentioned ambiguity in the factorization UY and the second is the fact that reasonable sparsity measures are not smooth and thus forbid the use of fast smooth optimization methods. We overcome these two problems by proposing an alternating minimization framework that gets rid of the ambiguity by iterating on a more appropriate geometric setting and allows us to use smooth approximations of h , which combine the advantage of having fast smooth optimization tools at hand together with the strong capability of non-convex sparsity measures in reconstruction tasks.

So let $h_\mu: \mathbb{R}^{m \times n} \rightarrow \mathbb{R}$ with $\mu \in \mathbb{R}^+$ be a smooth approximation of h such that h_μ converges pointwise to h as μ tends to zero. Monotonically shrinking the smoothing parameter μ between alternating minimization steps allows to combine the advantages of both smooth optimization and ℓ_0 -like sparsity measures. Schematically, we tackle problem (2) by iterating the following two steps until convergence.

Step #1 Let $L^{(i)} = U^{(i)} Y^{(i)}$ be the i -th iterate of the minimization process. In the first step, the estimate of the dominant subspace is improved by solving

$$U^{(i+1)} = \arg \min_{U \in \text{St}_{k,m}} h_\mu(\hat{X} - \mathcal{A}(UU^\top L^{(i)})). \quad (3)$$

We master the ambiguity problem by reformulating (3) as a minimization task on the set of symmetric rank- k projectors, which possesses a manifold structure and is known as the *Grassmannian*

$$\text{Gr}_{k,m} := \{P \in \mathbb{R}^{m \times m} \mid P = UU^\top, U \in \text{St}_{k,m}\}. \quad (4)$$

This results in the optimization problem

$$P^{(i+1)} = \arg \min_{P \in \text{Gr}_{k,m}} h_\mu(\hat{X} - \mathcal{A}(PL^{(i)})), \quad (5)$$

with $P^{(i+1)} = U^{(i+1)}(U^{(i+1)})^\top$.

Step #2 In the second step, the coordinates with respect to $U^{(i+1)}$ of the projected data points are adjusted by solving

$$Y^{(i+1)} = \arg \min_{Y \in \mathbb{R}^{k \times n}} h_\mu(\hat{X} - \mathcal{A}(U^{(i+1)}Y)). \quad (6)$$

Then, μ is decreased previous to the next iteration. Since ordinary PCA of the data allows a reasonably good initialization of U and Y , we initialize our algorithm with a truncated SVD of some matrix X_0 that is in accordance with the measurements, i.e. $\mathcal{A}(X_0) = \hat{X}$. The alternating scheme is summarized in Algorithm 1.

Algorithm 1 Alternating scheme for Robust PCA

Initialize:

Choose X_0 , s.t. $\mathcal{A}(X_0) = \hat{X}$.

Obtain $U^{(0)}$ from k left singular values of X_0 .

$$Y^{(0)} = U^{(0)\top} X_0$$

$$L^{(0)} = U^{(0)} Y^{(0)}$$

$$P^{(0)} = U^{(0)} U^{(0)\top}$$

Choose $\mu^{(0)}$ and $\mu^{(I)}$, compute

$$c_\mu = \left(\frac{\mu^{(I)}}{\mu^{(0)}} \right)^{1/(I-1)}$$

for $i = 1 : I$ **do**

$$P^{(i+1)} = \arg \min_{P \in \text{Gr}_{k,m}} h_{\mu^{(i)}}(\hat{X} - \mathcal{A}(PL^{(i)})) \quad \textbf{Step \#1}$$

$$\text{find } U^{(i+1)} \quad \text{s.t.} \quad U^{(i+1)} U^{(i+1)\top} = P^{(i+1)}$$

$$Y^{(i+1)} = \arg \min_{Y \in \mathbb{R}^{k \times n}} h_{\mu^{(i)}}(\hat{X} - \mathcal{A}(U^{(i+1)}Y)) \quad \textbf{Step \#2}$$

$$L^{(i+1)} = U^{(i+1)} Y^{(i+1)}$$

$$\mu^{(i+1)} = c_\mu \mu^{(i)}$$

end for

$$\hat{L} = U^{(I)} Y^{(I)}, \quad \hat{S} = X - \hat{L}$$

4 Optimizing the smooth sparsity measure

The proposed alternating minimization scheme involves the two non-convex but smooth optimization problems (5) and (6). We minimize both using a Conjugate Gradient type method due to its scalability and superlinear rate of convergence. The optimization methods are conceptually very similar but differ in the domains they operate on, as becomes clear from the cost functions and their respective gradients.

$$f_1 : \text{Gr}_{k,m} \rightarrow \mathbb{R}, \quad P \mapsto h_\mu(\hat{X} - \mathcal{A}(PL)), \quad \nabla f_1 = -\mathcal{A}^* \left(\nabla h_\mu(\hat{X} - \mathcal{A}(PL)) \right) L^\top \quad (7)$$

$$f_2 : \mathbb{R}^{k \times n} \rightarrow \mathbb{R}, \quad Y \mapsto h_\mu(\hat{X} - \mathcal{A}(UY)), \quad \nabla f_2 = -U^\top \mathcal{A}^* \left(\nabla h_\mu(\hat{X} - \mathcal{A}(UY)) \right) \quad (8)$$

Note that \mathcal{A}^* denotes the adjoint of the operator \mathcal{A} . Conjugate Gradient methods for minimization problems in the Euclidean case, such as (8) are standard and well established. In contrast to this, the core of our algorithm, i.e. minimizing (7) is a geometric optimization problem on the real Grassmannian and thus requires additional concepts such as vector transport and retraction.

In the following, we will recall some general concepts and further collect the ingredients for our algorithm. In particular, we derive a new retraction that is crucial for the algorithm's computational performance. For a deeper and more general insight into the topic of Geometric Optimization we refer to the work of Absil et al (2008).

4.1 Geometry of the Grassmannian

Consider a projector $P \in \text{Gr}_{k,m}$ as a point on the Grassmannian and let $\mathfrak{u}(m) := \{\Omega \in \mathbb{R}^{m \times m} | \Omega^\top = -\Omega\}$ be the set of skew-symmetric matrices. Using the Lie bracket operator $[Z_1, Z_2] = Z_1 Z_2 - Z_2 Z_1$, the set of elements in the tangent space of $\text{Gr}_{k,m}$ at P is given by $T_P \text{Gr}_{k,m} = \{[P, \Omega] | \Omega \in \mathfrak{u}(m)\}$. In the following, we will endow $\mathbb{R}^{m \times m}$ with the Frobenius inner product $\langle Z_1, Z_2 \rangle := \text{tr}(Z_1^\top Z_2)$ where $\text{tr}(\cdot)$ denotes the trace operator, and consider the Riemannian metric on $\text{Gr}_{k,m}$ accordingly as the restriction of this product to the respective tangent space. The orthogonal projection of an arbitrary point $Z \in \mathbb{R}^{m \times m}$ onto the tangent space at P is

$$\pi_P: \mathbb{R}^{m \times m} \rightarrow T_P \text{Gr}_{k,m}, \quad Z \mapsto [P, [P, Z_s]] \quad (9)$$

with $Z_s = \frac{1}{2}(Z + Z^\top)$ being the symmetric part of Z .

It is crucial for optimization procedures on manifolds to establish a relation between elements of the tangent space and corresponding points on the manifold, which motivates the usage of retractions. Conceptually, a retraction at some point P is a mapping from the tangent space at P to $\text{Gr}_{k,m}$ with a local rigidity condition that preserves gradients at P .

The generic way of locally parameterizing a smooth manifold is via Riemannian exponential mappings. As they are costly to compute in general we perform an approximation based on the QR -decomposition. Let $Gl(m)$ be the set of invertible $(m \times m)$ -matrices and let $\mathcal{R}(m) \subset Gl(m)$ be the set of upper triangular matrices with positive entries on the diagonal. It follows from the Gram-Schmidt orthogonalization procedure that the QR -decomposition $O(m) \times \mathcal{R}(m) \rightarrow Gl(m)$, $(Q, R) \mapsto QR$ is a diffeomorphism. Accordingly, every $A \in Gl(m)$ decomposes uniquely into $A =: A_Q A_R$ with $A_Q \in O(m)$ and $A_R \in \mathcal{R}(m)$. Moreover, it follows that the map

$$q_\Omega: \mathbb{R} \rightarrow O(m), \quad q_\Omega(t) := (I_m + t\Omega)_Q \quad (10)$$

is smooth for all $\Omega \in \mathfrak{u}(m)$ with derivative at 0 being $\dot{q}_\Omega(0) = \Omega$, cf. Kleinsteuber and Hüper (2007). Using

$$\alpha'_P: T_P \text{Gr}_{k,m} \rightarrow \text{Gr}_{k,m}, \quad \alpha'_P(\xi) := q_{[\xi, P]}(1) P (q_{[\xi, P]}(1))^\top \quad (11)$$

and exploiting the properties of the exponential mapping, one can develop the following result:

Lemma 1 *Consider arbitrary orthogonal matrices $\theta \in O(m)$. The mapping*

$$\alpha_{P,\theta}: T_P \text{Gr}_{k,m} \rightarrow \text{Gr}_{k,m}, \quad \alpha_{P,\theta}(\xi) := \theta (q_{\theta^\top [\xi, P] \theta}(1)) \theta^\top P \theta (q_{\theta^\top [\xi, P] \theta}(1))^\top \theta^\top \quad (12)$$

defines a set of retractions on $\text{Gr}_{k,m}$.

Proof 1 *The first condition, namely $\alpha_{P,\theta}(0) = P$, follows straightforwardly. It remains to show that $\frac{d}{dt} \big|_{t=0} \alpha_{P,\theta}(t\xi) = \xi$.*

Firstly, note that $q_{[\xi, P]}(1) = q_{[\xi, P]}(t)$. Since it has been verified that $\dot{q}_\Omega(0) = \Omega$ (cf. Helmke et al (2007)) and $[\xi, P]$ is skew symmetric, it follows that

$$\begin{aligned} \frac{d}{dt}\bigg|_{t=0} \alpha_{P, \theta}(t\xi) &= \theta \dot{q}_{\theta^\top[\xi, P]\theta}(0) \theta^\top P \theta (q_{\theta^\top[\xi, P]\theta}(0))^\top \theta^\top + \theta q_{\theta^\top[\xi, P]\theta}(0) \theta^\top P \theta (\dot{q}_{\theta^\top[\xi, P]\theta}(0))^\top \theta^\top \\ &= \theta \theta^\top [\xi, P] \theta \theta^\top P - P \theta \theta^\top [\xi, P] \theta \theta^\top \\ &= [P, [P, \xi]] \\ &= \xi \end{aligned}$$

To understand the last step of the equation, note that $\xi \in T_P \text{Gr}_{k,m}$ and therefore, ξ is invariant under the projection $\pi_P(\cdot)$. \square

As an associated vector transport, i.e. a mapping that for a given $\xi \in T_P \text{Gr}_{k,m}$ transports the tangent element $\eta \in T_P \text{Gr}_{k,m}$ along the retraction $\alpha_{P, \theta}(\xi)$ to the tangent space $T_{\alpha_{P, \theta}(\xi)} \text{Gr}_{k,m}$, we choose

$$\tau_{\xi, P, \theta}(\eta) := \theta (q_{\theta^\top[\xi, P]\theta}(1))^\top \theta^\top \eta (q_{\theta^\top[\xi, P]\theta}(1))^\top \theta^\top. \quad (13)$$

Note that in our algorithm the context of τ is always clear. Thus, we will drop the subscripts and simply write $\tau(\eta)$ for enhanced legibility.

4.2 CG on the Grassmannian

In the following, we sketch how the well-known nonlinear CG method extends to the Grassmannian for minimizing a smooth function $f: \text{Gr}_{k,m} \rightarrow \mathbb{R}$. Recall, that if f is the restriction of a smooth function $\hat{f}: \mathbb{R}^{m \times m} \rightarrow \mathbb{R}$, the Riemannian gradient in the tangent space is given by

$$\text{grad } f(P) = \pi_P(\nabla \hat{f}), \quad (14)$$

where $\nabla \hat{f}$ is the common gradient of \hat{f} in $\mathbb{R}^{m \times m}$. The CG method on the Grassmannian can be outlined as follows. Starting at an initial point $P^{(0)} \in \text{Gr}_{k,m}$, the Riemannian gradient $\Gamma^{(0)}$ can be computed and $H^{(0)} = -\Gamma^{(0)}$ is selected as initial search direction. In each iteration suitable step-size $t^{(i)}$ is determined using a backtracking line-search algorithm on the Grassmannian, cf. Algorithm 2. The new iterate is then obtained via $P^{(i+1)} = \alpha_{P^{(i)}}(t^{(i)} H^{(i)})$. Finally, the search direction

$$H^{(i+1)} = -\Gamma^{(i+1)} + \beta^{(i)} \tau(H^{(i)}) \quad (15)$$

is updated, where we consider two update rules for $\beta^{(i)}$, namely

$$\beta_{FR}^{(i)} = \frac{\langle \Gamma^{(i+1)}, \Gamma^{(i+1)} \rangle}{\langle \Gamma^{(i)}, \Gamma^{(i)} \rangle}, \quad \beta_{HS}^{(i)} = \frac{\langle \Gamma^{(i+1)}, (\Gamma^{(i+1)} - \tau(\Gamma^{(i)})) \rangle}{\langle \tau(H^{(i)}), (\Gamma^{(i+1)} - \tau(\Gamma^{(i)})) \rangle}. \quad (16)$$

The former is a Riemannian adaption of the well-known Fletcher-Reeves update formula and guarantees convergence of our algorithm (see the subsequent remark). The latter is an adaptation of the Hestenes-Stiefel formula and typically leads to better convergence behavior in practice, which is why we use it for all our algorithms.

Algorithm 2 Backtracking line search on Grassmannian

Choose $t_{init} > 0; c, \rho \in (0, 1)$ and set $t \leftarrow t_{init}$
repeat
 $t \leftarrow \rho t$
until $f(\alpha_P(tH^{(i)})) \leq f(P) + c t \text{tr}(\Gamma^{(i)\top} H^{(i)})$
Choose step-size $t^{(i)} := t$

Remark Convergence of the geometric CG method, which uses the retraction and vector transport as described above together with the Fletcher-Reeves update β_{FR} and the strong Wolfe-Powell condition, is guaranteed by a result of Ring and Wirth (2012) in the sense that $\liminf_{i \rightarrow \infty} \|\Gamma^{(i)}\| = 0$. Note, that the Wolfe-Powell condition would require a slight modification of Algorithm 2. However, we do not stress this issue since it has no impact on the practical convergence behavior.

4.3 Implementation of CG on Grassmannian

So far, the algorithm outlined above requires full $(m \times m)$ -matrices for the iterates $P^{(i)}$, $\Gamma^{(i)}$ and $H^{(i)}$. This is a drastic limitation on the performance of a practical implementation. In this section we derive a new retraction and show how it can be used to avoid full matrix multiplication and to reduce the storage requirements tremendously. The key idea is to decompose the projection matrices $P \in \text{Gr}_{k,m}$ into $P = UU^\top$ and to iterate on Stiefel matrices $U \in \text{St}_{k,m}$ instead. Moreover, one can exploit the structure of the tangent space $T_{\mathcal{I}} \text{Gr}_{k,m}$ at the standard projector \mathcal{I} , that is

$$\mathcal{I} = \begin{bmatrix} I_k & 0 \\ 0 & 0 \end{bmatrix}, \quad T_{\mathcal{I}} \text{Gr}_{k,m} = \left\{ \begin{bmatrix} 0 & A^\top \\ A & 0 \end{bmatrix} \mid A \in \mathbb{R}^{(m-k) \times k} \right\}. \quad (17)$$

Given U , a large QR -decomposition of U yields a fast way of constructing

$$V := \begin{bmatrix} U & U^\perp \end{bmatrix} \in O(m), \quad (18)$$

where $U^\perp \in \text{St}_{(m-k),m}$ denotes a basis of the orthogonal complement of the subspace spanned by U . Then, if $P = UU^\top$ and $\xi \in T_P \text{Gr}_{k,m}$, the identities

$$V^\top P V = \mathcal{I} \quad \text{and} \quad V^\top \xi V = \begin{bmatrix} 0 & A^\top \\ A & 0 \end{bmatrix} \quad (19)$$

hold for some $A \in \mathbb{R}^{(m-k) \times k}$. Therefore, instead of storing the full P and ξ it is sufficient to store U and A . Formally, this defines the bijection

$$\text{con}_V: T_P \text{Gr}_{k,m} \rightarrow \mathbb{R}^{(m-k) \times k}, \quad \text{con}_V(\xi) = A. \quad (20)$$

The projection onto $T_P \text{Gr}_{k,m}$ follows from a straightforward calculation.

Lemma 2 *Let $V = \begin{bmatrix} U & U^\perp \end{bmatrix}$, $P = UU^\top$ and $Z_s = \frac{1}{2}(Z + Z^\top)$ be defined as above and the orthogonal projection onto the tangent space at P be denoted by π_P . Then the identity*

$$\text{con}_V(\pi_P(Z)) = (U^\perp)^\top Z_s U \quad (21)$$

holds.

Proof 2 *Using the definition of $\pi_P(Z)$ and the fact that $V^\top P V = \mathcal{I}$,*

$$\begin{aligned} V^\top \pi_P(Z) V &= [V^\top P V, V^\top [P, Z_s] V] \\ &= [\mathcal{I}, [\mathcal{I}, V^\top Z_s V]] \end{aligned}$$

and the same structure as in (19) is obtained. \square

Lemma 1 allows to choose a particular retraction that is easy to compute. Consider $A = \text{con}_V(\xi)$ and let

$$A = \theta_A \begin{bmatrix} R \\ 0 \end{bmatrix} \quad (22)$$

be the large QR -decomposition of A , with $\theta_A \in O(m-k)$ and R an upper triangular (not necessarily invertible) $(k \times k)$ -matrix. Furthermore, define

$$M(R) := \begin{bmatrix} I_k & -R^\top \\ R & I_k \end{bmatrix} \quad (23)$$

and its Q -factor $\theta_M \in O(2k)$.

Lemma 3 *Let $\alpha_{P,\theta}(\xi)$ be a retraction as in (12), where θ is chosen as $\theta := V \begin{bmatrix} I_k & 0 \\ 0 & \theta_A \end{bmatrix}$. Then the Stiefel matrix*

$$\widetilde{U} := \theta \begin{bmatrix} \theta_M & 0 \\ 0 & I_{m-2k} \end{bmatrix} \begin{bmatrix} I_k \\ 0 \end{bmatrix} \in \text{St}_{k,m} \quad (24)$$

satisfies $\alpha_{P,\theta}(\xi) = \widetilde{U}\widetilde{U}^\top$.

Proof 3 *One can deduce $q_{\theta^\top[\xi,P]\theta}(1) = (I_m + \theta^\top[\xi,P]\theta)_Q = \begin{bmatrix} \theta_M & 0 \\ 0 & I_{m-2k} \end{bmatrix}$ from*

$$\begin{aligned} \theta^\top[\xi,P]\theta &= \begin{bmatrix} I_k & \theta_A^\top \\ 0 & \theta_A^\top \end{bmatrix} [V^\top \xi V, V^\top P V] \begin{bmatrix} I_k & 0 \\ 0 & \theta_A \end{bmatrix} \\ &= \begin{bmatrix} I_k & 0 \\ 0 & \theta_A^\top \end{bmatrix} \begin{bmatrix} 0 & -A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} I_k & 0 \\ 0 & \theta_A \end{bmatrix} \\ &= \begin{bmatrix} 0 & -A^\top \theta_A \\ \theta_A^\top A & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & [-R^\top & 0] \\ [R] & 0 \end{bmatrix}. \end{aligned}$$

Therefore, the retraction is

$$\begin{aligned} \alpha_{P,\theta}(\xi) &= \theta \begin{bmatrix} \theta_M & 0 \\ 0 & I_{m-2k} \end{bmatrix} \theta^\top P \theta \begin{bmatrix} \theta_M^\top & 0 \\ 0 & I_{m-2k} \end{bmatrix} \theta^\top \\ &= V \begin{bmatrix} I_k & 0 \\ 0 & \theta_A \end{bmatrix} \begin{bmatrix} \theta_M & 0 \\ 0 & I_{m-2k} \end{bmatrix} \underbrace{\begin{bmatrix} I_k & 0 \\ 0 & \theta_A^\top \end{bmatrix} V^\top P V \begin{bmatrix} I_k & 0 \\ 0 & \theta_A \end{bmatrix}}_{\mathcal{I}} \begin{bmatrix} \theta_M^\top & 0 \\ 0 & I_{m-2k} \end{bmatrix} \begin{bmatrix} I_k & 0 \\ 0 & \theta_A^\top \end{bmatrix} V^\top \\ &= V \begin{bmatrix} I_k & 0 \\ 0 & \theta_A \end{bmatrix} \begin{bmatrix} \theta_M & 0 \\ 0 & I_{m-2k} \end{bmatrix} \begin{bmatrix} I_k \\ 0 \end{bmatrix} \begin{bmatrix} I_k & 0 \end{bmatrix} \begin{bmatrix} \theta_M^\top & 0 \\ 0 & I_{m-2k} \end{bmatrix} \begin{bmatrix} I_k & 0 \\ 0 & \theta_A^\top \end{bmatrix} V^\top, \end{aligned}$$

which completes the proof. \square

Similarly to the retraction, the vector transport from (13) is simplified, as is described in the following.

Lemma 4 Let $\theta, \xi, P, \tilde{U}$ be as above and $\tilde{V} := [\tilde{U} \mid \tilde{U}^\perp]$. Then for $\eta \in T_P \text{Gr}_{k,m}$ and $B := \text{con}_V(\eta)$, the identity $\text{con}_{\tilde{V}}(\tau_{\xi,P,\theta}(\eta)) = \theta_A^\top B$ holds.

Proof 4 From (19) we have $V^\top \eta V = \begin{bmatrix} 0 & B^\top \\ B & 0 \end{bmatrix}$. The vector transport can thus be written as

$$\tau_{\xi,P,\theta}(\eta) = \theta \begin{bmatrix} \theta_M & 0 \\ 0 & I_k \end{bmatrix} \begin{bmatrix} I_k & 0 \\ 0 & \theta_A^\top \end{bmatrix} \begin{bmatrix} 0 & B^\top \\ B & 0 \end{bmatrix} \begin{bmatrix} I_k & 0 \\ 0 & \theta_A \end{bmatrix} \begin{bmatrix} \theta_M^\top & 0 \\ 0 & I_k \end{bmatrix} \theta^\top.$$

Using the fact that

$$\theta^\top [\tilde{U} \mid \tilde{U}^\perp] = \theta^\top \left[\theta \begin{bmatrix} \theta_M & 0 \\ 0 & I_{m-2k} \end{bmatrix} \begin{bmatrix} I_k \\ 0 \end{bmatrix} \mid \theta \begin{bmatrix} \theta_M & 0 \\ 0 & I_{m-2k} \end{bmatrix} \begin{bmatrix} 0 \\ I_{m-k} \end{bmatrix} \right] = \begin{bmatrix} \theta_M & 0 \\ 0 & I_{m-2k} \end{bmatrix}$$

we can show that

$$\begin{aligned} \tilde{V}^\top \tau_{\xi,P,\theta}(\eta) \tilde{V} &= [\tilde{U} \mid \tilde{U}^\perp]^\top \theta \begin{bmatrix} \theta_M & 0 \\ 0 & I_{m-2k} \end{bmatrix} \begin{bmatrix} I_k & 0 \\ 0 & \theta_A^\top \end{bmatrix} \begin{bmatrix} 0 & B^\top \\ B & 0 \end{bmatrix} \begin{bmatrix} I_k & 0 \\ 0 & \theta_A \end{bmatrix} \begin{bmatrix} \theta_M^\top & 0 \\ 0 & I_{m-2k} \end{bmatrix} \theta^\top [\tilde{U} \mid \tilde{U}^\perp] \\ &= \begin{bmatrix} I_k & 0 \\ 0 & \theta_A^\top \end{bmatrix} \begin{bmatrix} 0 & B^\top \\ B & 0 \end{bmatrix} \begin{bmatrix} I_k & 0 \\ 0 & \theta_A \end{bmatrix} \\ &= \begin{bmatrix} 0 & B^\top \theta_A \\ \theta_A^\top B & 0 \end{bmatrix} \end{aligned}$$

and $\text{con}_{\tilde{V}}(\tau_{\xi,P,\theta}(\eta))$ follows again from the matrix structure. \square

Algorithm 3 illustrates how the minimization of (2) on the Grassmannian is efficiently implemented in practice. Note, that therein, H and G denote the preimages of the respective bijection con_V and thus are of dimension $(m-k) \times k$. We do not further discuss the CG method in the Euclidean space as it is a standard method. In all minimization procedures, the convergence is evaluated by observing the progress in decreasing the cost function.

Algorithm 3 Implementation of CG on $\text{Gr}_{k,m}$

input $U = U^{(i)}$

Obtain V and U^\perp from Eq. (18)

Compute $G = (U^\perp)^\top \nabla \tilde{f}(UU^\top)U$

$H = -G$

repeat

Obtain θ_H, R from H as in Eq. (22)

Determine step-size t acc. Algorithm 2

Obtain θ_M from QR -dec. of $M(tR)$ (23)

Update U according to (24)

Update V, U^\perp, G as above

Compute $\tau(G) = \theta_H^\top G$ and $\tau(H) = \theta_H^\top H$

Update H following (15) and (16)

until converged

output $U^{(i+1)}$

5 Robust subspace tracking

In this section, we extend the aforementioned mathematical tools to robustly track the underlying subspace. To that end, we require our sparsity measure h to be separable, meaning that the sparsity measure of a matrix $A = [a_1, a_2, \dots, a_n]$ consists of the sum of sparsity measures of its columns a_i . Note, that all sparsity measures given in Eq. (34) fulfill that condition. By slight abuse of notation, we write

$$h\left(\begin{bmatrix} A & a \end{bmatrix}\right) = h(A) + h(a), \quad (25)$$

where $a \in \mathbb{R}^m$ is the last column of the matrix $\begin{bmatrix} A & a \end{bmatrix}$. Assume now that the current observation matrix $\hat{X}^{(i)}$ is updated by a new observation vector $\hat{x}^{(i+1)}$, leading to a new observation matrix $\hat{X}^{(i+1)} = \begin{bmatrix} \hat{X} & \hat{x} \end{bmatrix}$. Following (2), the new optimization problem is

$$\min_{U \in \text{St}_{k,m}, Y \in \mathbb{R}^{k \times n}, y \in \mathbb{R}^k} h\left(\begin{bmatrix} \hat{X} & \hat{x} \end{bmatrix} - \mathcal{A}\left(U \begin{bmatrix} Y & y \end{bmatrix}\right)\right). \quad (26)$$

We show how the above problem can profit from knowledge of the optimal subspace in the previous iteration. Due to the separability of h and the properties of \mathcal{A} we can separate the current observation from the previous optimization problem by rewriting

$$h\left(\begin{bmatrix} \hat{X} & \hat{x} \end{bmatrix} - \mathcal{A}\left(U \begin{bmatrix} Y & y \end{bmatrix}\right)\right) = h(\hat{X} - \mathcal{A}(UY)) + h(\hat{x} - \mathcal{A}(Uy)). \quad (27)$$

Furthermore, we introduce a weighting factor $0 < w < 1$ for incoming observations, which will play the role of a *forgetting factor* in the actual tracking algorithm. The optimization problem for subspace tracking now reads as

$$\min_{U \in \text{St}_{k,m}, Y \in \mathbb{R}^{k \times n}, y \in \mathbb{R}^k} (1 - w)h(\hat{X} - \mathcal{A}(UY)) + wh(\hat{x} - \mathcal{A}(Uy)). \quad (28)$$

Assume $P^{(i)}$ and $L^{(i)} = U^{(i)}Y^{(i)}$ that minimize $h(\hat{X} - \mathcal{A}(L))$ and an initial estimate $l_0 = P^{(i)}x_0$ with $\mathcal{A}(x_0) = \hat{x}^{(i+1)}$ are available for the current data vector. Then a two-step update rule similar to (5) and (6) can be formulated. In a first step, update the dominant subspace estimate

$$P^{(i+1)} \approx \arg \min_{P \in \text{Gr}_{k,m}} (1 - w)h(\hat{X} - \mathcal{A}(PL^{(i)})) + wh(\hat{x} - \mathcal{A}(Pl_0)) \quad (29)$$

via gradient descent on $\text{Gr}_{k,m}$ and decompose $P^{(i+1)} = U^{(i+1)}U^{(i+1)\top}$. Note that the gradient $\Gamma^{(i+1)}$ can be updated from the previous $\Gamma^{(i)}$ and the current observation. Then adjust the coordinates of the current low-rank estimate by optimizing

$$y^{(i+1)} = \arg \min_{y \in \mathbb{R}^k} (1 - w)h(\hat{x} - \mathcal{A}(U^{(i+1)}y)). \quad (30)$$

While (30) can be solved in the same fashion as before by employing a CG method in Euclidean space, the solution of problem (29) is approximated by one gradient descent step on the Grassmannian, i.e.

$$P^{(i+1)} = \alpha_{P^{(i)}}(\beta \Gamma^{(i+1)}) = \alpha_{P^{(i)}}(\beta((1 - w)\Gamma^{(i)} + w\gamma^{(i+1)})). \quad (31)$$

In this formula, β is a suitable step size from standard gradient descent methods,

$$\Gamma^{(i)} = \text{grad}_P h(\hat{X} - \mathcal{A}(PL^{(i)})) \quad (32)$$

is the Riemannian gradient at the preceding iteration, and

$$\gamma^{(i+1)} = \text{grad}_P h(\hat{x} - \mathcal{A}(Pl^{(i+1)})) \quad (33)$$

the Riemannian gradient for the current observation. The tracking scheme is illustrated in Alg. 4.

Algorithm 4 Subspace tracking

Initialize:

Select initial data set \hat{X}_0 , consisting of at least k data vectors.

Estimate $U^{(0)}$ and $Y^{(0)}$ following the alternating minimization outlined in Alg. 1

$$L^{(0)} = U^{(0)} Y^{(0)}$$

$$P^{(0)} = U^{(0)} U^{(0)\top}$$

Compute $\Gamma^{(0)}(\hat{X}_0, P^{(0)}, L^{(0)})$ according to (32)

for each new sample $\hat{x} = \hat{x}^{(i+1)}$ **do**

Choose w and μ as well as x_0 , s.t. $\mathcal{A}(x_0) = \hat{x}$.

Initialize $y_0 = U^{(i)\top} x_0$ and $l_0 = U^{(i)} y_0$

Compute $\gamma^{(i+1)}(x_0, P^{(i)}, l_0)$ according to (33)

Update $\Gamma^{(i+1)} = (1 - w)\Gamma^{(i)} + w\gamma^{(i+1)}$

Update $P^{(i+1)} = \alpha_{P^{(i)}}(\beta\Gamma^{(i+1)})$ cf. (29), (31)

find $U^{(i+1)}$ s.t. $U^{(i+1)} U^{(i+1)\top} = P^{(i+1)}$

$y^{(i+1)} = \arg \min_{y \in \mathbb{R}^k} h_\mu(\hat{x} - \mathcal{A}(U^{(i+1)}y))$ cf. (30)

$$l^{(i+1)} = U^{(i+1)} y^{(i+1)}$$

end for

To further reduce computational cost, the following lemma proves useful.

Lemma 5 Define $V = \begin{bmatrix} U & U^\perp \end{bmatrix}$ and the Riemannian Gradients Γ and γ as above and denote by $G = (U^\perp)^\top \Gamma U$ and $g = (U^\perp)^\top \gamma U$ their respective preimages according to (20). Then the identity

$$\text{con}_V((1 - w)\Gamma + w\gamma) = (1 - w)G + wg$$

holds.

The proof follows from the linearity of (20).

Using this result, the computational effort reduces tremendously, because explicit computation of $\gamma^{(i+1)}$ and $\alpha_{P^{(i)}}(\beta\Gamma^{(i+1)})$ is not required for the new iterate $U^{(i+1)}$. Considering (7) for the case of a single data vector, γ and with it g become rank-one matrices. Thus, it is sufficient to compute G and its QR -decomposition in the initialization phase and then to perform lightweight rank-one updates (cf. Golub and Van Loan (1996)) on the Q and R factors to update G and U in each iteration. The detailed procedure is illustrated in Algorithm 5.

Algorithm 5 Implementation of the Gradient descent update

input $x_0, l_0, P^{(i)}, V = \begin{bmatrix} U^{(i)} & | & U^{\perp(i)} \end{bmatrix}, \theta_G^{(i)}$ and $R^{(i)}$ from the QR -decomposition of $G^{(i)}$ as in (22)

Compute $\gamma(x_0, P^{(i)}, l_0)$ and $g = \text{conv}_V(\gamma)$

Find u, v s.t. $uv^\top = g$

Obtain $\theta_G^{(i+1)}$ and $R^{(i+1)}$ via rank-one-update from $\theta_G^{(i)}, R^{(i)}, u$ and v

Compute $\theta = V \begin{bmatrix} I_k & 0 \\ 0 & \theta_G^{(i+1)} \end{bmatrix}$

Determine optimum step-size t from line-search along $H = -G^{(i+1)}$

Obtain θ_M from QR -decomposition of $M(tR^{(i+1)})$ (23)

Update U according to (24)

output $U^{(i+1)}, \theta_G^{(i+1)}, R^{(i+1)}$

6 Experiments and Evaluation

This section gives an overview of the actual implementation and the performance of the presented algorithms. Firstly, we refer to practical issues such as the selection of suitable penalty functions and the choice of parameters. In the following, we evaluate the performance of our approach in the Robust PCA task and demonstrate how the proposed method outperforms other state-of-the-art algorithms. Concluding the experiments, we illustrate the method's behavior on real-world data by tracking a low-dimensional subspace in a visual background reconstruction task.

6.1 Smooth penalty functions

Inspired by the work of Gasso et al (2009), we investigate the following smoothed ℓ_0 -surrogates:

$$h_\mu^{\text{lp}}: \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^+, \quad X \mapsto \sum_{j=1}^n \sum_{i=1}^m \left(x_{ij}^2 + \mu \right)^{\frac{p}{2}}, \quad 0 < p < 1 \quad (\text{lpnorm}) \quad (34a)$$

$$h_\mu^{\text{log}}: \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^+, \quad X \mapsto \sum_{j=1}^n \sum_{i=1}^m \log \left(1 + \frac{x_{ij}^2}{\mu} \right) \quad (\text{logarithm}) \quad (34b)$$

$$h_\mu^{\text{atan}}: \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^+, \quad X \mapsto \sum_{j=1}^n \sum_{i=1}^m \text{atan}^2 \left(\frac{x_{ij}}{\mu} \right) \quad (\text{atan}) \quad (34c)$$

Using this kind of cost function instead of e.g. the ℓ_1 -norm has several advantages. The smoothing parameter μ can be tuned in a way that the cost function either penalizes large outliers (similar to the Frobenius norm) or enforces sparsity on the outliers regardless of their magnitude. Within a practical implementation it is observed that larger values of μ lead to faster convergence of the optimization algorithm while small values, as expected, lead to much sparser residuals. We profit from this flexibility by adjusting the smoothing parameter and thus the resulting sparsity. Specifically, in the subspace reconstruction task we prefer faster convergence in the beginning in order to quickly obtain a reliable rough estimate of the subspace and reduce μ after each alternation. In the limit $\mu \rightarrow 0$ the cost functions behave similarly as the ℓ_0 -norm, which leads to the best results, as the following experiments demonstrate.

6.2 Numerical experiments for the subspace reconstruction task

In order to have a quantitative measure for the recovery performance of the robust PCA algorithms, we conducted numerical experiments with artificial test data and employed the available ground truth for a quantitative evaluation. We compare our algorithm to *SpaRCS* (Waters et al, 2011), *RPCA* (*EALM* and *IALM*) (Lin et al, 2010), *GRASTA* (He et al, 2012) and *GoDec* (Zhou and Tao, 2011), and use MATLAB implementations provided by the authors. In all experiments we constructed the test data $X \in \mathbb{R}^{m \times n}$ to be the sum of a matrix L of fixed rank k (with $k < m$) and a sparse matrix S . The low-rank component is obtained by computing the singular value decomposition $U\Sigma V^\top$ of a zero-mean, $\mathcal{N}(0, 1)$ -distributed random matrix, assigning zero to all singular values $\sigma_i, i = k+1 \dots m$ and reconstructing the matrix $F = U\tilde{\Sigma}V^\top$. In order to control the magnitude of the entries for varying k we scale the entries of F to a unit sample standard deviation, obtaining a normalized $L = \frac{1}{\text{std}(F)}F$. The entries of S are randomly placed and uniformly distributed in $[-5, 5]$, which makes a well-proportioned relation between data points and outliers. Some other comparisons follow the data model of Candes et al (2009), where the low-rank matrix is the product of two $\mathcal{N}(0, \frac{1}{n})$ -distributed random matrices. The entries of S follow a Bernoulli distribution and thus surpass the amplitude of the data points by several orders. In our evaluation the outliers cannot be detected just from their magnitude, but they are still large enough to severely distort the estimated subspace.

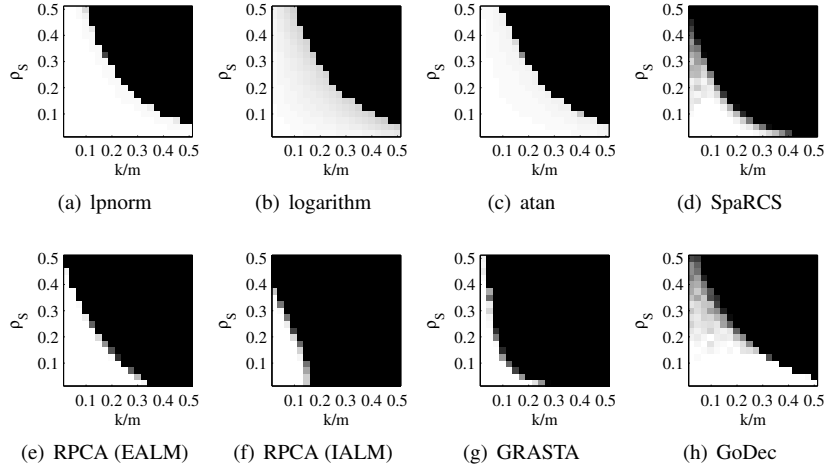


Figure 1: Subspace reconstruction performance measured by phase transitions in rank and sparsity (dark area: reconstruction failed). Our method (a)-(c) achieves the highest reconstruction rate.

A widely used performance measure for subspace reconstruction are the phase transitions in rank and sparsity. A data set of dimension $m = n = 400$ is generated and the subspace recovery is evaluated while varying both the relative rank k/m and the relative sparsity $\rho_S = \|S\|_0/mn$ in the range of 0.025 to 0.5. The reconstruction is considered abortive if $\frac{\|L - \hat{L}\|_F}{\|L\|_F} > 0.05$.

The results in Figure 1 demonstrate that the proposed framework (a)-(c) using ℓ_0 -surrogates covers a broad range of scenarios and surpasses *SpaRCS*, *GRASTA* and the

RPCA methods in the subspace reconstruction task. Especially in the cases of either k/m being very small and ρ_S being very large or vice versa, our algorithm is still able to recover the subspace while other methods fail. Out of all compared methods, the *GoDec* algorithm comes closest to our performance - however, it has to be stressed that we must feed the exact cardinality of S into the *GoDec* algorithm, which is not available in a real-world application. We chose the number of iterations between the alternating minimization steps as $I = 50$ for all phase transition plots. We choose $p = 0.5$ and shrink μ from 0.9 to 10^{-4} for (34a), from 2 to 0.005 for (34b) and, respectively, from 2 to 0.05 for (34c).

6.3 Incomplete observations

In a second experiment we evaluate our algorithm for the case of data being incompletely observed and compare the performance against *SpaRCS*, which is especially designed for compressive measurements. The results in Figure 2 illustrate that our method recovers a broader range of configurations than the competing *SpaRCS*, which has its strengths especially for very low-rank matrices. In general, with less samples being available, the lower the dimension of the underlying subspace and the sparser the outliers have to be for a successful reconstruction. We stress that (i) the outliers are placed at random positions and (ii) the data is subsampled in a completely random way.

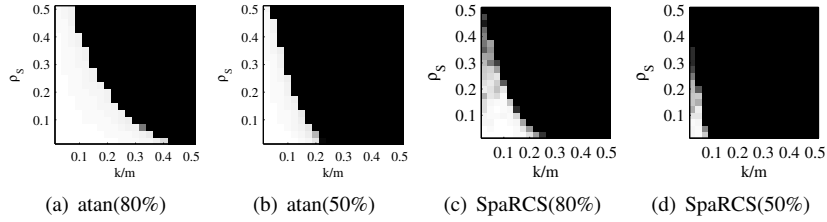


Figure 2: Phase transitions at recovering a low-rank matrix from incomplete observations

6.4 Noisy reconstruction

To investigate the behavior of the proposed algorithm in the presence of additive Gaussian noise we perform the following experiments: ρ and k/m are fixed to 0.1 and a Gaussian noise term $N \in \mathbb{R}^{m \times n}$ of a particular energy level is added to the data. We run our alternating minimization method for 10 iterations and compare the performance and the runtime against the above mentioned competing algorithms. Intuitively, a low-rank and sparse decomposition on a noisy dataset should mostly affect the sparse component, as Gaussian noise is full rank and only a small amount of noise will affect the low-rank component. Figure 3 illustrates the recovery precision for the low-rank matrix, where the SNR measures the relation between the energy of L and N . As the results reveal, the proposed rank-controlling method provides a rather noise-robust estimation of the subspace, which supports our choice of a simple model. Like all other methods, the reconstruction quality deteriorates with increasing noise level. However,

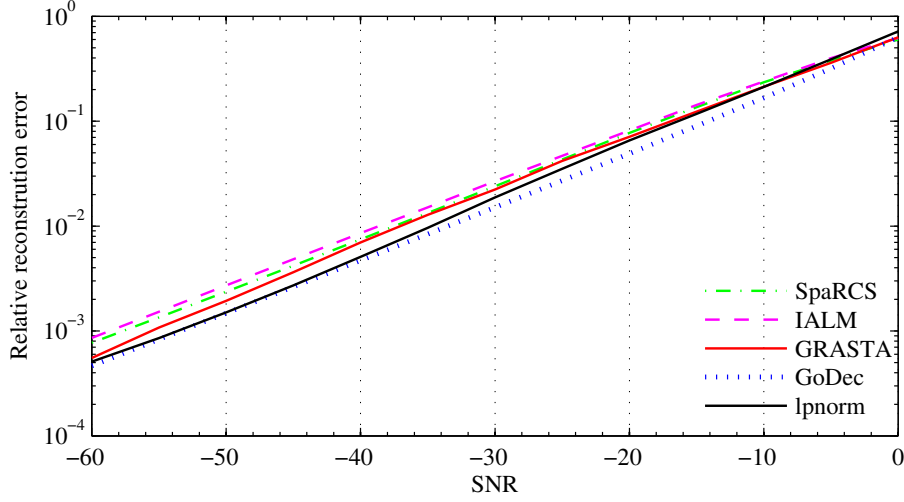


Figure 3: Relative subspace recovery error at different Signal-to-Noise-Ratios

at a moderate noise level our algorithm performs equally well as *GoDec*, which in contrast to our method requires additional knowledge about the cardinality of the sparse component.

Concerning the average runtime for solving the noisy decomposition task on a desktop computer in MATLAB, *GoDec* performs fastest in 1.5 seconds and the *IALM* method requires about 2.7 seconds. Our framework needs an average runtime between 4.2 (*atan*) and 5.3 (*l1norm*) seconds to solve the task and outperforms *SpaRCS*, which requires 6.1 seconds. Lastly, *EALM* and *GRASTA* are rather costly with 30 and, respectively, 111 seconds.

6.5 Subspace tracking on a real-world example

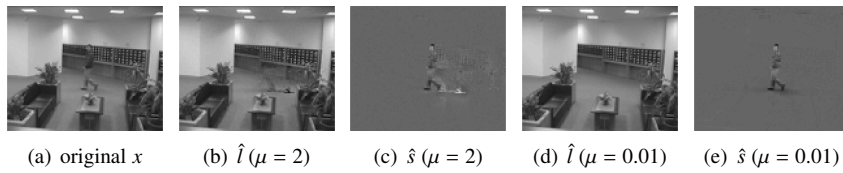


Figure 4: Subspace tracking - Background and foreground for different smoothing parameters

For the purpose of robustly tracking the underlying subspace, we selected the dataset *lobby* from Li et al (2004) and perform the widely popular background subtraction task. In this task, the background of a pixel-wisely sampled video is modeled by a low-rank approximation and foreground objects can automatically be extracted as they are assumed to be sparse. The selected scene shows an office scenario with people occasionally walking through the scene. It is especially challenging as the lighting conditions change significantly after about 400 frames. We initialize our tracking al-

gorithm with the first 50 frames of the sequence using the proposed alternating minimization method for 10 steps and perform our tracking algorithm on the subsequent frames. From our ℓ_0 -surrogates we choose (34c) and upper-bound the desired rank k by 2. To demonstrate the influence of the smoothing parameter we fix $\mu = 2$ for one experiment and for the other we shrink μ from 2 to 0.01 in the initialization phase and leave it for the following tracking procedure. The choice of a suitable weighting parameter is a trade-off between reaction time (i.e. how long it takes to adapt the subspace) and the risk of overfitting the subspace (i.e. $\hat{l} \approx x$), which leads to ghost images in the reconstruction. For all experiments we select $w = 0.05$ as a weighting factor.

The sample observation of frame # 365 in Figure 4 illustrates that in both configurations the subspace \hat{l} is successfully recovered, as the sparse foreground estimates $\hat{s} = x - \hat{l}$ (4(c) and (e)) contain only the moving person in the scene. However, the smaller μ is selected, the less blurry the extracted silhouette, as becomes clear from the comparison between the two experiments. The more sparsity on the residual is enforced, the more effectively can ghost images from previous observations be suppressed in the extracted foreground objects.

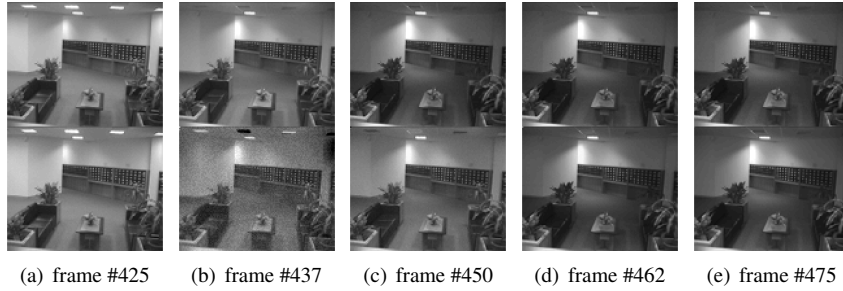


Figure 5: Tracking a change in room illumination. Upper: original video, lower: low-rank approximation ($\mu = 0.01$)

Figure 5 illustrates the behavior of the algorithm when a change occurs in the background. Although the lighting conditions following frame # 425 have not been observed during the initialization, our tracking algorithm adjusts the subspace to the new background within a small number of frames.

7 Conclusion

We present a framework for Robust PCA that is able to recover a low-rank matrix from a data set corrupted by sparse outliers and missing data. Experiments show that our method covers a wider range of scenarios in terms of higher rank and greater number of outliers than other state of the art methods. Even if the data is incompletely observed a comparably good performance is achieved. The same holds true for data sets corrupted by additive Gaussian noise.

Instead of tackling a convex relaxation of the problem, our algorithm reaches as closely as possible to the ideal Robust PCA objective by modelling the problem as simply as possible and making no additional assumptions, e.g. on the number of sparse outliers. Instead of the widely-popular ℓ_1 -measure we propose using ℓ_0 -surrogates, which can be adjusted so that the optimization is performed rather smoothly or, in the

limit, sparsity is enforced in an ℓ_0 -like behavior, which leads to superior performance. In order to obtain an inherent upper bound on the dimension of the dominant subspace we propose an optimization problem on the Grassmannian and derive an efficient retraction that saves time and storage and makes the algorithm applicable for large-scale problems. Furthermore we show how the method can be adapted to allow tracking a subspace that varies over time and use real-world data to demonstrate the performance.

References

- Absil PA, Mahony R, Sepulchre R (2008) Optimization Algorithms on Matrix Manifolds. Princeton University Press, Princeton, NJ
- Balzano L, Nowak R, Recht B (2010) Online identification and tracking of subspaces from highly incomplete information. In: Communication, Control, and Computing, 2010 48th Annual Allerton Conference on, pp 704–711
- Boumal N, Absil PA (2011) RTRMC: A Riemannian trust-region method for low-rank matrix completion. In: Advances in Neural Information Processing Systems, pp 406–414
- Cai J, Candès EJ, Shen Z (2010) A singular value thresholding algorithm for matrix completion. *SIAM J on Optimization* 20:1956–1982
- Candès E, Li X, Ma Y, Wright J (2009) Robust principal component analysis? *Journal of ACM* 58(3):1–37
- Chartrand R, Staneva V (2008) Restricted isometry properties and nonconvex compressive sensing. *Inverse Problems* 24(3):1–14
- Chen Y, Xu H, Caramanis C, Sanghavi S (2011) Robust Matrix Completion and Corrupted Columns. In: International Conference on Machine Learning, vol 2, pp 873–880
- Ding C, Zhou D, He X, Zha H (2006) R1-PCA: Rotational invariant L1-norm Principal Component Analysis for robust subspace factorization. In: 23rd international conference on Machine learning, ACM, New York, NY, USA, pp 281–288
- Gasso G, Rakotomamonjy A, Canu S (2009) Recovering Sparse Signals With a Certain Family of Nonconvex Penalties and DC Programming. *IEEE Transactions on Signal Processing* 57(12):4686–4698
- Golub H, Van Loan CF (1996) Matrix computations. Johns Hopkins University Press, Baltimore, MD, USA
- He J, Balzano L, Szlam A (2012) Incremental gradient on the Grassmannian for online foreground and background separation in subsampled video. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 1568–1575
- Helmke U, Hüper K, Trumpf J (2007) Newton’s method on Grassmann manifolds. *ArXiv e-prints* 0709.2205
- Keshavan RH, Montanari A (2010) Matrix completion from noisy entries. *The Journal of Machine Learning Research* 11:2057–2078

- Kleinstauber M, Hüper K (2007) An Intrinsic CG Algorithm for Computing Dominant Subspaces. In: IEEE International Conference on Acoustics, Speech and Signal Processing, 7, pp 1405–1408
- Kwak N (2008) Principal component analysis based on ℓ_1 -norm maximization. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(9):1672–80
- Li L, Huang W, Gu IYH, Tian Q (2004) Statistical modeling of complex backgrounds for foreground object detection. Image Processing, IEEE Transactions on 13(11):1459–1472
- Lin Z, Chen M, Wu L, Ma Y (2010) The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. Arxiv preprint arXiv:1009.5055 arXiv:1009.5055v2
- Meyer G, Bonnabel S, Sepulchre R (2011) Linear regression under fixed-rank constraints: A Riemannian approach. In: International Conference on Machine Learning, pp 545–552
- Ring W, Wirth B (2012) Optimization methods on Riemannian manifolds and their application to shape space. SIAM Journal on Optimization 22(2):596–627
- Shalev-Shwartz S, Gonen A, Shamir O (2011) Large-Scale Convex Minimization with a Low-Rank Constraint. In: International Conference on Machine Learning, pp 329–336
- Shalit U, Weinshall D, Chechik G (2010) Online learning in the manifold of low-rank matrices. Advances in Neural Information Processing Systems 23:2128–2136
- Waters A, Sankaranarayanan AC, Baraniuk RG (2011) SpaRCS: Recovering Low-Rank and Sparse Matrices from Compressive Measurements. In: Advances in Neural Information Processing Systems
- Wright J, Ganesh A, Rao S, Peng Y, Ma Y (2009) Robust Principal Component Analysis: Exact Recovery of Corrupted Low-Rank Matrices via Convex Optimization. In: Advances in Neural Information Processing Systems, pp 2080–2088
- Zhou T, Tao D (2011) GoDec: Randomized low-rank & sparse matrix decomposition in noisy case. In: International Conference on Machine Learning, pp 33–40